Agile User Stories Enriched with Usability

Ana M. Moreno and Agustín Yagüe

Universidad Politecnica de Madrid
Madrid, Spain
ammoreno@fi.upm.es, agustin.yague@upm.es

Abstract. Usability is a critical quality factor. Therefore, like traditional software teams, agile teams have to address usability to properly catch their users experience. There exists an interesting debate in the agile and usability communities about how to achieve this integration. Our aim is to contribute to this debate by discussing the incorporation of particular usability recommendations into user stories, one of the most popular artifacts for communicating agile requirements. In this paper, we explore the implications of usability for both the structure of and the process for defining user stories. We discuss what changes the incorporation of particular usability issues may introduce in a user story. Although our findings require more empirical validation, we think that they are a good starting point for further research on this line.

1 Introduction

ISO 9241-11 [1] defines usability as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specific context of use". In short, usability is also generally referred to as "quality in use" [2].

The integration and cross pollination between usability and agile practices have been a rapidly expanding area of work and research in the last few years. The increasing number of publications concerning the field or the active Yahoo discussion group called Agile Usability are two signs of change. One of the premises of this line of work is that usability is a critical quality factor and needs to be dealt with during agile development in order to provide a quality user experience. Both the HCI and agile communities agree on this point. On the HCI side, for example, Nielsen [3] states that an agile development team must recognize interaction design and usability methods as explicit methodology development components, whereas, on the agile side, Ambler [4] claims that an end product's good usability can be ensured only by systematic usability engineering activities during development iterations.

This is not, however, a straightforward process. Different authors have highlighted challenges that need to be overcome if both fields want to work together. Differences in terminology (Ferreira et al.[5]), goals (Lee[6]) and approaches to software construction (Desilets[7]) are some of the most often cited obstacles to this integration.

Nonetheless, several topics dealing with this road to integration are under debate. At the organizational level, there is an interesting discussion about how the UX team should work with the agile team (Ferreira et al.[8]). Another interesting line of work addresses when UX design should take place in an agile process (Constantine[9]), [10,11,12,13].

Some time ago, the HCI literature provided very specific usability recommendations with a clear positive impact on the final quality of use of software systems. Examples are give the user the option to cancel an ongoing process [11,12,13,14], to undo a task [15,16], provide the user feedback on what is going on in the system [15,17,18], adapt software functionalities to the user profile [19] or provide clear and marked exits for the application [17]. Such usability recommendations are in line with what Nielsen lately referred to as fast and cheap usability techniques [20], as quick usability actions that help to significantly increase user satisfaction.

Such recommendations represent specific functionalities to be incorporated into a software system. Therefore, as discussed in [21], they can be considered as functional usability requirements that complement traditional requirements.

Advancing along the above road to usability and agile integration, we address how to deal with the above functional usability requirements in an agile context. We explore how to represent such functional usability requirements in user stories, one of the most popular artifacts for conveying agile requirements.

To do this, we have structured the paper as follows. Section 2 describes the usability recommendations that we will deal with and discusses the need for full specification. Then Section 3 discusses an approach for documenting this type of usability information into user stories. Section 4 introduces a software tool set up to support the inclusion of usability mechanisms in user stories. Section 5 describes how the approach is validated. Finally, Section 6 outlines some conclusions and future work.

2 Specifying Functional Usability Features

This section describes the usability recommendations that we will deal with and discusses the need for full specification. We have worked on the functional usability recommendations proposed in [21], that is, usability heuristics with key benefits (according to the usability literature) and with strong design implications, according to the software engineering literature. Table 1 provides an overview.

One question that arises is whether such features need to be explicitly specified and, if so, exactly what information should be listed. For example, would it be enough to state in the user story that a particular functionality should include status feedback? From a usability perspective, many details have to be taken into account for a system to provide satisfactory *system status feedback*, including what states to report, what information to display for each state, how prominent the information should be in each case (for example, should the application keep control of the system or should users be able to do something else while system status is being reported)... Therefore, much more information than just a description of the usability feature must be considered in order to properly build such feedback recommendations into a software system.

Table 1. Usability mechanisms addressed

Usability Mechanism	Goal
System Status	To inform users about the internal status of the system
Warning	To inform users of any action with important consequences
Long Action Feedback	To inform users that the system is processing an action that will take some time to complete
Global Undo	To undo system actions at several levels
Abort Operation	To cancel the execution of an action or the whole application
Abort Command	To cancel the execution of a task in progress
Go Back	To go back to a particular state in a command execution sequence
Structured Text Entry	To help prevent the user from making data input errors
Step-by-Step Execution	To help users to do tasks that require different steps with user input and correct such input
Preferences	To record each user's options for using system functions
Favorites	To record certain places of interest for the user
Multilevel Help	To provide different help levels for different users

Notice that neither customers/users, nor, as Chamberlain et al. [13] claim, agile developers are generally usability experts. So, unless this type of usability information is documented in some way, good usability would, as Jokela and Abrahamsson [22] mentioned, be more or less a fluke resulting from customer and/or developer intuition. Whether the sources of this information are customers/users, developers, usability experts or usability elicitation guidelines [21], such information may, from an agile perspective, require new user stories and/or modifications to the original functional stories (new acceptance criteria, new tasks…). Therefore, they will have an impact on the workload associated with the respective user stories and, consequently, on the sprint plan. It is our understanding that this type of usability information should be somehow represented or documented as part of user stories, so it can be properly estimated and implemented. The next section discusses an approach for documenting usability information in user stories.

3 Documenting Usability in User Stories

Bearing in mind recommendations on how to write good user stories [23] and documentation on usability mechanisms [24], we have identified three ways in which the incorporation of usability influences user stories:

1. Addition of new stories to represent requirements directly derived from usability. We call these new stories "usability stories" to distinguish them from traditional user stories, as they represent usability features to be provided by the system.
2. Addition or modification of tasks in existing user stories. This means that some actions derived from usability constraints should be performed in an existing user story. This task could be as simple or detailed as needed.
3. Addition or modification of acceptance criteria. These acceptance criteria appear because the user story functionality needs to include some specific actions that modify the operating environment.

At least one, if not all three, of these three actions has to be taken when writing user stories with usability. Table 2 shows the implications of each analyzed usability mechanism when it is included in a user story. Table columns represent the above actions and rows contain the usability mechanisms . Cells marked with an "X" signal that the incorporation of the usability mechanism requires the respective action. For example, the implementation of the warning mechanism affects the user story by modifying acceptance criteria, adding new acceptance criteria, adding new tasks and adding a new usability story to the product backlog. Table 2 was built empirically as a result of two case studies and is being further validated, as discussed later.

Table 2. Mapping between usability mechanisms and actions

	New Task	Modify Task	New Acceptance Criteria	Modify Acceptance Criteria	New Usability Story	New User Story
System Status		X	X	X	X	
Warning	X		X	X	X	
Long Action		X	X		X	
Abort command	X	X	X		X	
Abort operation		X	X			
Go Back	X	X	X			
Text entry		X	X			
Step by Step	X		X	X		
Preferences						X
Favorites		X	X		X	
Help		X	X		X	

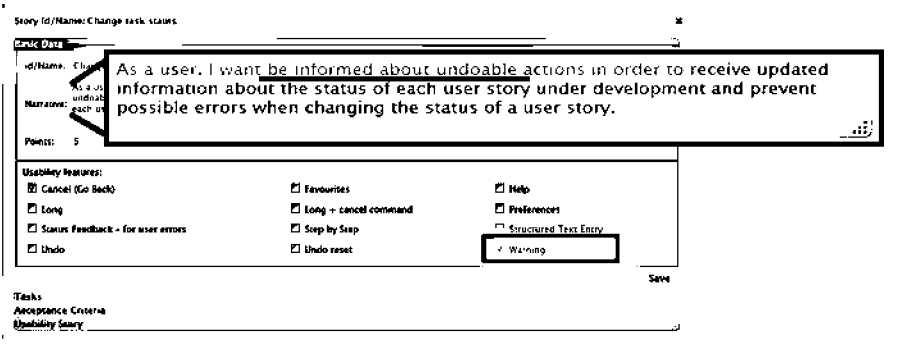

Fig. 1. User story description with usability features

Based on [23,25], the term usability story could be defined as "an artifact that is used to represent usability features that a system/software should support because they are needed by a user to use in a more easy and trusty way and that gives value to the user/acquirer. Usability stories are documented as user stories because both are similar. The next section shows an example of a usability story for implementing

warning messages. User stories and usability stories are referred to differently to highlight that usability stories are created to address usability requirements related to a particular user story. Usability stories will be elicited not from product owners but from usability mechanisms designed to improve the use of a particular functionality represented in a user story. The next section gives an example of a user story including the warning usability mechanism.

4 Tool and Process

To support the inclusion of usability mechanisms into user stories, we have modified an open source tool for managing user stories (ScrumTime http://www.scrumtime.org/). The main features added to ScumTime are:

- A list of usability mechanisms available as checkbox items to be associated with each user story.
- A list of usability affected tasks: when a usability mechanism has been selected for a user story, recommendations about new tasks to be added (or the modifications to existing tasks) as a result of including this mechanism are displayed in the task panel.
- A list of usability affected acceptance criteria: when a usability mechanism has been selected for a user story, the new criteria (or changes to existing criteria) to be taken into account to check that the implementation covers the usability features are displayed in the acceptance criteria panel.
- Usability story: when a usability mechanism has been selected for a user story and this mechanism requires the creation of a usability story, the usability story is automatically added to the product backlog.
- Help functionality: examples on how to add usability tasks and acceptance criteria for each usability feature are provided through a new help functionality.

Let us look at an example to illustrate how the tool works. Consider an application managing user stories in agile projects. One of the features of this application might be "*graphically change the status (created, in progress, stopped, done) of a user story*". A user story description that does not consider usability features might read "**As a user, I want to change the status of a user story and receive updated information about the status of each user story under development**".

This user story description does not include any information about usability. Suppose, for example, that customers want to be warned about undoable actions (*warning feature*). Following the process described by authors in [26], this feature should be added to the user story because some technical actions have to be taken to inform customers. Fig. 1shows how the tool does this. The description has been zoomed-in and the words related to the warning pattern have been highlighted.

Basic tasks and acceptance criteria are fixed later, when the user story is detailed during sprint planning. A new task has to be added to account for the usability feature.

As Fig. 2 shows, a new task, highlighted by a zoomed-in black box, is added to describe the warning task. All the tasks that are required because of the usability features are listed on the right side of the screen. They are also boxed in black.

Finally, new acceptance criteria should also be considered (see Fig. 3.), plus a new usability story to represent the warning window.

If different usability mechanisms are associated with a user story, the implications for the tasks and acceptance criteria will appear in a compressed folder which users can expand at their convenience (Figs. 2 and 3 show the go back and warning mechanisms, for example).

The main objective of the tool is to capture the usability meta-knowledge related to the inclusion of particular usability mechanisms in a software system. Consequently, a developer without too much usability knowledge can contribute to the development of usable systems. Notice that, as already discussed; neither users nor developers are ordinarily usability experts. Therefore, an automated tool storing this usability meta-knowledge can be helpful if there are no usability experts on hand.

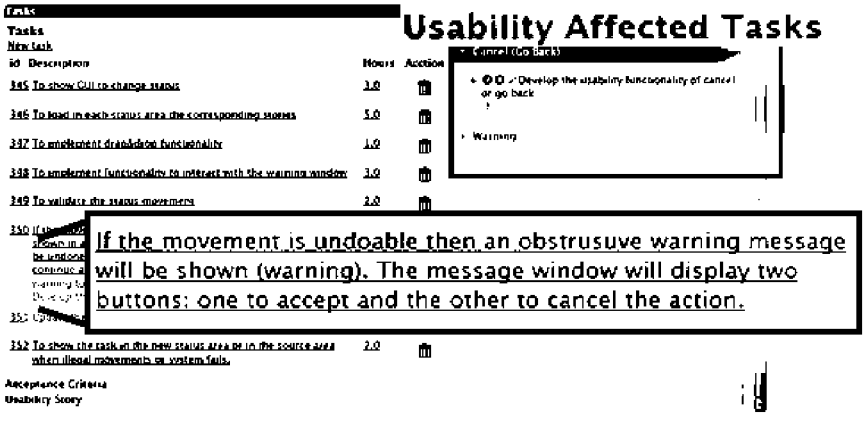

Fig. 2. User story tasks that support the warning feature

5 Proof of Concept

At the time of writing this paper, the validation process was still in progress. It is, however, a two-part process. First, we have worked on validating the usability knowledge summarized in Table 2. To do this, UPM software engineering graduate students developed a small agile project (a tool for managing user stories) as part of their degree project. The tool implemented 24 usability stories incorporating the described usability mechanisms. Using the results we were able to test the usability implications for tasks, acceptance criteria and new usability stories derived from each usability mechanism.

Second, we are validating the tool described in Section 4. It has been tested by UPM software engineering master students, all of whom have 2 to 4 years' experience as software practitioners. As part of their master's thesis, they developed a real application using our tool for creating and documenting user stories. In particular, we worked with three agile teams composed of 3 to 4 developers. The final results are still under evaluation, but early feedback suggests that usability features were easy to add to the user stories and not much usability knowledge was required to do so. The main identified problems were management issues concerning the removal of usability features after their tasks or acceptance criteria were defined.

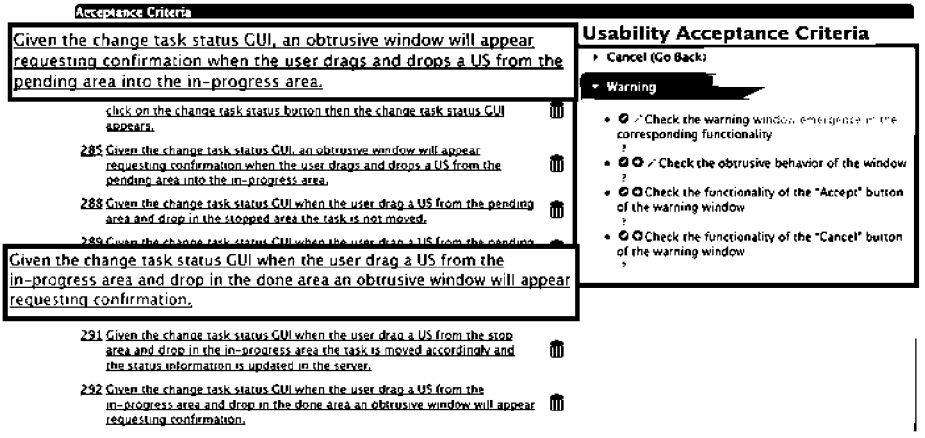

Fig. 3. User story acceptance criteria with usability

Finally, from the preliminary experience using the tool, we can conclude that it is easy to check the usability features of each user story, but it takes some practice to incorporate the usability discussion into the regular user story creation flow. The tool is available at *http://scrumtime.eui.upm.es*.

6 Conclusion

Our hypothesis is that usability constraints may have a major impact on the system to be built. They should, therefore, be dealt with in the development process. This paper aims to present preliminary results on incorporating particular usability mechanisms into agile user stories.

We map the main usability mechanisms and their implications for user stories and also introduce a tool that captures the usability knowledge related to such implications. The approach is still undergoing validation, but preliminary results suggest that the workload for incorporating particular usability mechanisms using the stored usability knowledge leads is reasonably acceptable.

The concept of usability story has been defined to represent the stories needed to implement the required usability mechanisms.

This research raises several issues, like, for example, when to deal with usability functionalities in an agile process or how to manage the size of user stories containing quite a few of usability mechanisms.

The next steps are related to further validating our solution and a detailed analysis of the open issues.

References

1. ISO 9241-11, 98: Ergonomic Requirements for office work with Visual Display Terminals. Part 11: Guidance on Usability. ISO (1998)
2. ISO/IEC. 1999, ISO14598-1, 99: Software Product Evaluation: General Overview. ISO/IEC (1999)
3. Nielsen, J.: Agile Development Projects and Usability. Jakob Nielsen's Alertbox, November 17 (2008), http://www.useit.com/alertbox/agile-methods.html (visited December 2010)
4. Ambler, S.W.: Tailoring Usability into Agile Software Development Projects. In: Law, E., Hvannberg, E., Cockton, G. (eds.) Maturing Usability. Quality in Software, Interaction and Value. Springer, Heidelberg (2008)
5. Ferreira, J., Noble, J., Biddle, R.: Agile development iterations and UI design. In: AGILE 2007: Proc. of the AGILE 2007, pp. 50–58. IEEE Computer Society, Washington, DC (2007)
6. Lee, J.C.: Embracing Agile Development of Usable Software Systems. In: CHI (2006)
7. Desilets, A.: Are Agile Usability and Methodologies Comparable (2005), http://www.carleton.ca/hotlab/hottopics/Articles/June2005-AreAgileandUxMet.html (visited on December 2010)
8. Ferreira, J., Sharp, H., Robinson, H.: Values and Assumptions Shaping Agile Development and User Experience Design in Practice. In: Sillitti, A., Martin, A., Wang, X., Whitworth, E. (eds.) XP 2010. LNBIP, vol. 48, pp. 178–183. Springer, Heidelberg (2010)
9. Constantine L.L.: Process agility and software usability: Toward lightweight usage-centered design. Constantine & Lockwood, Ltd., Tech. Rep. 110 (2001), http://citeseer.ist.psu.edu/465732.html
10. Miller, L.: Case study of customer input for a successful product. In: ADC 2005: Proceedings of the Agile Development Conference, pp. 225–234. IEEE Computer Society, Washington, DC, USA (2005)
11. Patton, J.: Hitting the Target: Adding Interaction Design to Agile Software Development. In: Proceedings of OPSLA 2004 (2004)
12. Haikara, J.: Usability in Agile Software Development: Extending the Interaction Design Process with Personas Approach. In: Concas, G., Damiani, E., Scotto, M., Succi, G. (eds.) XP 2007. LNCS, vol. 4536, pp. 153–156. Springer, Heidelberg (2007)
13. Maiden, N., Chamberlain, S., Sharp, H.: Towards a Framework for Integrating Agile Development and User-Centred Design. Springer, Heidelberg (2006)
14. Usability Pattern Collection (December 2010), http://www.cmis.brighton.ac.uk/research/patterns/
15. Shneiderman, B.: Designing the User Interface: Strategies for Effective Human-Computer Interaction. Addison-Wesley (1998)
16. Tidwell, J.: Designing Interfaces. In: Patterns for Effective Interaction Design. O'Reilly (2005)
17. Nielsen, J.: Usability Engineering. John Wiley & Sons (1993)
18. van Welie, M.: Patterns in Interaction Design, http://www.welie.com (accessed November 2008)
19. Rubinstein, R., Hersh, H.: The Human Factor. Digital Press, Bedford (1984)
20. Nielsen, J.: Fast, Cheap, and Good: Yes, You Can Have It All (January 2007), http://www.useit.com/alertbox/fast-methods.html (visited December 2010)

21. Juristo, N., Moreno, A., Sanchez-Segura, M.-I.: Guidelines for eliciting usability functionalities. IEEE Trans. Softw. Eng. 33(11), 744–758 (2007)
22. Jokela, T., Abrahamsson, P.: Usability Assessment of an Extreme Programming Project: Close Co-operation with the Customer Does Not Equal to Good Usability. In: Bomarius, F., Iida, H. (eds.) PROFES 2004. LNCS, vol. 3009, pp. 393–407. Springer, Heidelberg (2004)
23. Cohn, M.: User Stories Applied: For Agile Software Development. The Addison-Wesley Signature Series. Addison-Wesley Professional (March 2004), http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20n&path=ASIN/0321205685
24. Juristo, N., Moreno, A.M., Sanchez-Segura, M.-I.: Analysing the impact of usability on software design. J. Syst. Softw. 80(9), 1506–1516 (2007)
25. Beck, K.: Extreme Programming Explained: Embrace Change. Addison Wesley (1999)
26. Moreno, A.M., Yague, A.: Adding usability recommendations into Agile user stories. In: Proc. 1st Workshop Dealing with Usability in an Agile Domain at XP 2010 (2010)